Day5 5-4-20

index.html

```
<body>
  <app-root> </app-root>
</body>
```

app.module.ts

```
import{ BrowserModule} from '@angular/platform-browser';
import{ NgModule} from '@angular/core';
import{ AppRoutingModule} from './app-routing.module';
import{ AppComponent}
import{ StudentComponent} from './student/student.component
import{ MyserService} from './myser.service';
@NgModule({
 declarations:[AppComponent, StudentComponent],
 imports:[BrowserModule, APPRoutingModule],
 provider: [MyserService].
 bootstrap:[ AppComponent].
})
```

app.component.ts

```
import{Component, Inject} from '@angular/core';
import {MyserService} from './myser.service';
@Component({
   selector: 'app-root',
   templateUrl: './app.component.html',
   styleUrls: ['./app-component.css']
})
 export class AppComponent
 {    title = 'hello App';
      constructor(@Inject(MyserService)ref_myser)
      {
         console.log(ref_myser)
      }
   }
```

myser.service.ts

```
   import{Injectable} from '@angular/core';
   @Injectable({
         providedIn: 'root'
   })
   export class MyserService{
      constructor()
      {
      }
   }
```

Output:-

app component app.

console screen

```
hello all are in modulet
▼ MyserServic{ }
 ▼ _proto_:
  ▶ constructor: class MyserService
  ▶ _proto_ : object.
```

Creating one component

index.html

```
<body>
 <app-root></app-root>
<body>
```

app.component.ts

```
import{Component, HostListener}from '@angular/core';
@Component({
selector : 'app-root'
templateUrl : './app.component.html',
 styleUrls : ['./app.component.css']
})
export class AppComponent{
     title = "helloApp";
}
```

app.component.html

```
<h1> app component app </h1>
<app-student> </app-student>
```

student.component.css

```
```

student.component.ts

```
import{Component, OnInit}from @angular/core';
@component({
 selector: 'app-student'
templateUrl: './student.component.html'
 styleUrls : ['./student.component.css']
})
export class StudentComponent implements
    OnInit{
          constructor(){ }
          ngOnInit(){
    }
}
```

student.component.html

```
<h1> we are in student component
          html file</h1>
<p> student works </p>
```

Adding two components

app.component

app.component.ts

```
import{component} from '@angular/core';
@Component({
 selector:'app-root'
 templateUrl: './app.component.html';
 styleUrls:['./app.component.css]
})
export classAppComponent {
 title="helloApp';
}
```

app.component.css

```
h1
{ color:red
}
```

app.component.html

```
<h1> in app component.html</h1>
<p> is template </p>
<app-employee></app-employee>
<app-student> </app-student>
```

employee componnt

student compont

employee.component.ts

```
import{Component} from'@angular/cor'
@Component({
 selector: 'app-employee'
templateUrl: './employee.component.html
 styleUrls:['./employee.component.css]
})
```

employee.component.css

```
h1
{
 color:red
}
```

employee.component.html

```
<h1> Employee.component.html
                </h1>
<p> employee works </p>
```

~~emplo~~student.component.ts

```
import{Component} from '@angular/core'
@Component({
selector: 'app-student'
template Url: './student.component.html'
 styleUrls: ['./student.component.css]
})
```

student.component.css

```
h1
{
 color:red
}
```

~~stu~~student.component.html

```
<h1> student component.html
                   </h1>
<p> Student works! </p>
```

Note :-

```
declarations : [
    AppComponent
    StudentComponent  }  Updated
    EmployeeComponent }  automatically.
]
```

Output :-

```
It is app.component.app
it is template.
In Employee.component.html
employee works!
we are in student.component.html file
student works!.
```

app.component.css

```
h1
{
    color: blue
}
```

student.component.css

```
h1
{
    color: green
}
```

employee.component.css

```
h1
{
    color: red
}
```

Output :-

It is app compontent app → Blue color
it is template
In Employee component.html → Red color
employee works!
We are in student.component.html file → Green color.
student works!

Global styling :-

here we will make use of styles.css

app.component.css

```
h1
{
  color: red;
}
```

employee.component.css

```
h1
{
  color: red;
}
```

student.componentcs

```
h1
{
  color: red;
}
```

styles.css

```
h1
{
  color: red;
}
```

If there are multiple of component in which h1 tag is present we cannot do styling to each components css file.
-to-a

- In order to avoid this we will make use of global style page i.e., styles.css.

Output:-

It is app.component.app
It is template.

In employee.component.html
employee works

We are in student.component.html.file
student works!

Inline html (Inline template)

- If only one line of code is present then we can make use of single code (' ')
- If there is multi-line html code then we should make use of backtick (` `)

- Inline html or inline styling is not recommendable for multiple line of codes. If we write multiple line in app.component.ts then the code will be increased so we have to avoid writing inline style or inline template.

Inline Template

app.component.ts

```
import{component} from '@angular/core';
@Component({
  selector: 'app-root'
  template:
        '<h1>we are in the app component</h1>
          <p> It is inline template</p>'
  stylesUrls: ['/app.component.css']
})
export class AppComponent
{     title = 'heloAPP';
}
```

Inline Style

app.component.tss

```
import{Component} from '@angular/core';
 @Component({
  selector: 'app-root'
  template Url: '<h1>we are in the app component</h1>
                 <p> it is inline template </p>'

  styles: [' h1{
                 color: red
                 }
               p{
                 color: blue
                }'
          ]
 })
 export class AppComponent{
       title: "heloApp"
 }
```

* Inline css has more priority and if you want to apply css property to one or two lines then we will make use of In line- styling.